APPENDIX

[NB: The term "playback" should be construed to refer to any rendering of the content to outputs, including archiving to file, broadcasting, etc. It is simply more convenient to use the word "playback".]

I. Media base APIs and functionality

Both this section and the following will discuss APIs, categorized by type of functionality exposed.

a. Opening/closing multimedia

i. OpenURL: Far and away the most common way of opening multimedia using the Media base, this will cause the Media base to use its source resolver to create a Media Source that will be able to read the content at the specified URL.

ii. OpenSource: If an application has already created a Media Source and would like the presentation to use that source, he would call OpenSource(), and the Media base would use that source directly.

iii. OpenBindable: This is for applications that do not have a Media Source but instead have an object that implements IBindable from which a Media Source object could be obtained. The IBindable interface is from a Device Directory.

iv. OpenByteStream: This is for applications that have an IMFByteStream from which the multimedia data to be presented can be read. IMFByteStream is the Media Base abstraction for any object from which sequential data can be obtained.

v. OpenTopology: Advanced applications may wish to create the topology defining the sources, sinks, and transforms to be used in the presentation, instead of relying on the Media base and Media Session to do it (see III.a.iii). These applications can supply the topology and bypass any source- or topology-resolution that would otherwise be done.

vi. Close: Closes the current media. The Media base can subsequently be reused for new media by using any of the above "Open" calls.

b. Presentation control

i. Start: The application calls this to start playback. The parameters allow the application to specify a start position and an end position for playback, with defaults that request playback to start at the beginning and play until the end. There is a third parameter that defines the units of these start and end parameters. The default units are 10MHz time units, but other systems of specifying positions in a multimedia presentation – such as frame number or time codes, for example – can be used.

Start can also be called while the Media base is already started; this seeks to a new location in the content and starts playback from there.

ii. Stop: The application calls this to stop playback. . The time shown on the clock will go back to zero. If the application calls Start() again with default parameters, playback will start at the beginning of the presentation.

iii. Pause: The application calls this to pause playback. The time shown on the clock will remain "frozen". If the application calls Start() again with default parameters, playback will resume from where it was paused.

iv. SetPresentationTimeSource: This is for advanced applications that wish to drive the time on the clock for the presentation. If the application calls this, then all components will attempt to run according to the time exposed by this time source. In the absence of this call, the Media Session selects a time source from among all components that offer one (see III.b.iii)

c. Information querying

The following APIs are useful for applications that wish to obtain more information about the presentation.

i. GetCapabilities: This method exposes various capabilities of the Media base and Media Session that can change during a

presentation.

ii. GetMetadata: This method allows the application to obtain a property store which can be queried for any metadata concerning the presentation (e.g. duration, title, author, etc)

iii. GetDestination: Retrieves a pointer to the destination in use

iv. GetStatistics: Retrieves statistics on the Media base's activities

d. Shutdown: This is called once when the Media base will no longer be used. It releases all resources and references to other components (including Media Sessions), shutting them down as well.

e. Events

The Media base will notify the application of events via its media event generator. The following is a list of some of the more common events, although other events generated by the components inside the Media Session will also be propagated to the application.

i. MEMediaOpened: This event is accompanied by a presentation descriptor describing how the output of the presentation will look.
ii. MEMediaStarted
iii. MEMediaStopped
iv. MEMediaPaused
v. MEMediaEnded
vi. MEMediaClosed
vii. MEMediaRateChanged
viii. MEEngineStateChanged
ix. MENewPresentation: This event is accompanied by a presentation descriptor describing how the presentation coming out of the Media Source(s) will look. The event will be received once after one of the "Open" calls is made, and once for every new presentation that the Media Source will output subsequently in timeline scenarios. Applications can optionally use this information to configure the destination in anticipation of this new presentation.
x. MEPresentationSwitched: This event is sent once the new presentation is actually playing back. It reflects all of the various outputs in which the presentation is occurring.
xi. MEOutputsUpdated: This event is sent in the case of a destination (output) change, once the change has been fully

resolved and put in place. It reflects all of the various outputs in which the presentation is now occurring.

II. Media Session APIs and functionality

Every presentations that uses the Media Base control layer is run by a Media Session that is instantiated or otherwise obtained by the Media base. Although Media Base provides a Media Session implementation that will be used for the vast majority of scenarios, other Media Sessions that expose the below API can be built according to specification. This is generally useful if the multimedia being presented does not lend itself to concepts of Media Sources, Media Sinks, and other MF constructs that would be used in the MF-provided Media Session. For instance, a Media Session capable of running Shockwave Flash presentation will be provided in Media Base.

a. Session configuration

The following APIs allow the Media base to configure the Media Session for a presentation

i. ResolveTopology: The Media base calls this when it has created a partial topology for the presentation (see III.a.iii) that it needs resolved into a full topology.

ii. SetTopology: The Media base calls this once it has a full topology for the upcoming presentation. The Media Session sets this on the Media Processor, which sets up the pipeline to get data from the Media Source through all of the transforms specified in the topology.

iii. SetPresentationClock: When this is called, the Media Session ensures that all components (Media Sinks and some Media Sources) subscribe to receive notifications from the clock, which will be used to control the presentation.

iv. SetConfigurationPropertyStore: An extensible set of configuration parameters is passed to the Media Session using this method.

b. Presentation control

The following APIs allow the Media base to control the presentation. For II.b.i through II.b.iv, see the comments in the Media base section

above (I.b)

i. Start
ii. Stop
iii. Pause
iv. SetPresentationTimeSource

v. Preroll: The Media base uses this to tell the Media Session to get everything ready for the start of an upcoming presentation

c. Information querying

These APIs allow the Media base to get information from the Media Session

i. GetSessionGUID: Identifies the implementation of the Media Session. As mentioned above, Media Base provides an implementation that is used in the vast majority of scenarios, but other implementations can be used as well.
ii. GetSessionCapabilities. Similar to I.c.i.

d. Shutdown
e. Events

Events generated by the Media Session are received by the Media base. For some of them, the Media base translates them into one of the events listed in the list of Media base events (I.e). For som, the Media base acts on the information contained therein and does not propagate them to the application. For others, the Media base propagates them to the application.

i. MESessionPrerolled
ii. MESessionStarted
iii. MESessionStopped
iv. MESessionEnded
v. MESessionPaused
vi. MEMediaRateChanged

III. Presentation flow in Media Base

This section is an overview of a typical multimedia scenario, along with a description of what the Media base and Media Session do to drive the presentation(s).

a. Media base work

i. Source resolution

This step obtains a Media Source from which the multimedia data can be read. This step is relevant when OpenURL or OpenByteStream are used to open the multimedia. The other Open functions specify the Media Source directly.

The Media base passes the URL or the Byte Stream to the Source Resolver. If the Source Resolver was given an URL, then it looks at the scheme of the URL (file://, http://, etc) to create a Byte Stream that will read from the specified location.

In both cases, the Source Resolver is able to use at the contents of the Byte Stream to determine the format of the bits (ASF, AVI, MEPG, etc) so that a Media Source can be instantiated that will understand that format.

ii. Setting up the Media Session

The Media Source is asked for a presentation descriptor. This descriptor may specify that a custom Media Session is to be used. This is not usually the case, though; usually, the default Media Base Media Session is instantiated here.

iii. Partial topology resolution

The Media base obtains a presentation descriptor from the Media Source and notifies the application of that presentation via MENewPresentation. If the application is interested in handling that event, the Media base waits for the application to finish handling it.

The Media base then negotiates with the application-provided destination and creates Media Sinks for the outputs of the presentation. The Media base constructs a "partial topology", which indicates the source Media Streams and the output Stream Sinks, without necessarily specifying the transforms that will be needed to get there.

iv. Topology resolution and activation

The Media base asks the Media Session to resolve the partial topology into a fully-specified topology. Then the Media base sets the new fully-specified topology on the Media Session.

v. Presentation control

The application now can control the progress of the presentation by calling Start, Stop, and Pause on the Media base. The Media base forwards these calls to the Media Session, which handles them.

vi. Handling of new presentations from the Media Source

If the Media Source is a timeline, then it will notify the Media Session of upcoming new presentations by means of an event, which will forward the event to the Media base. The Media base then goes through steps III.a.iii through III.a.iv using a descriptor for the new presentation. Once playback of that new presentation is about to commence, MEPresentationSwitch is sent to the application.

vii. Handling of output changes

If the Media base receives an event from the application-provided Destination notifying it of a change on the output side, it goes through steps III.a.iii through III.a.iv using a descriptor for the existing presentation. Once playback using the new outputs is about to commence, MEOutputsUpdated is sent to the application.

b. Media Session work

The below details how the main Media Base implementation of the Media Session works. Other Media Sessions must provide the APIs and functionality listed above (II), but they may accomplish it in a different manner.

i. Full topology resolution

The Session resolves the topology using the Topology Loader, which figures out what transforms are necessary to get data from the source Media Streams to the output Stream Sinks.

ii. Media Processor creation

The Session owns the Media Processor. When the topology is set on the Media Session, the Media Session sets the topology on the Media Processor, which configures a pipeline from the Media Streams to the formats needed by the Stream

Sinks.

iii. Time source selection

Upon starting the presentation, the Media Session makes the determination of which of the available time sources will be used to drive the presentation. Each component will run its part of the presentation in sync with the time from this time source.

Media Sinks may optionally offer a time source. Typically, the audio renderer will do this, and the time on the time source will be dictated by the audio device on the machine, but other Media Sinks may do so as well. Media Source may also offer time sources. The Media Session decides which of these is the "highest priority" time source, and this time source is used by the main presentation clock, to which all clock-aware components sync their presentations.

iv. Presentation control

The Media Session will get calls to Preroll, Start, Stop, and Pause from the Media base. These typically correspond to the applications calls on the Media base.

The Media Session will control the presentation via the Presentation Clock that the Media base gave it. Since all Media Sinks, starting/stopping/pausing the Presentation Clock will result in all sinks receiving notifications thereof and reacting appropriately. The Media Session starts/stops/pauses the Media Processor by calling its Start/Stop/Pause methods directly.

The Media Session will send an event to the Media base after a given operation has been completed by all streams.

v. Bit pumps

The Media Session drives all data flow from the Media Processor's streams to the Stream Sinks. Each Media Sink - Stream Sink pair is associated with a Bit Pump that, while it is in the started or prerolling state, operates in the following loop:

1. Request a sample allocation from the Stream Sink
2. When the sample allocation request is filled, request that the Media Stream fill the sample with data

3. When the sample-filling request is filled, hand the sample to the Stream Sink, and request another sample allocation.

   Note that more than one sample can be "in the air" at any given time.

vi. Handling of new presentations and output changes

See III.a.vi and III.a.vii above. The Media Session is responsible for forwarding the Media Processor's notification of the upcoming presentation to the Media base and participating with topology resolution and activation.

IV. Minimum Media base configuration required

Far and away, the most common use of Media Base will be to set up simple playback of a multimedia presentation. As noted in the "strategic importance" section above, an important goal of Media Base is to make it very easy to set this up.

From the application's point of view, the following steps describe what an application must, at minimum, do in order to configure a multimedia presentation using Media Base:

a. Create a Media base
b. Create a playback destination, providing a handle to the window in which the video for the presentation should be rendered.
c. Call IMFMediaEngine::OpenURL, supplying an URL to the multimedia file to be presented, as well as a pointer to the playback destination.
d. The media presentation can now be played back, using IMFMediaEngine::Start/Stop/Pause APIs. Note that the application does not need to wait for any events to arrive; handling of these events are optional.

V. Provision of additional services

Advanced multimedia applications will take advantage of a wide and extensible array of services made available by the Media base and Media Session. These are accessed by using the IMFGetService interface exposed by the Media base, which will return services exposed by the Media base, the Media Session, or any one of the Media Sources, Sinks, or other components that are in use.

This service architecture is extensible, but the following are a few

examples:

a. Rate support and rate control
b. Volume control for audio
c. Frame caching support for editing scenarios
d. Video renderer configuration